CLAIMS:

1. A method of executing processes with different priorities in a multiprocessing environment comprising execution of a low priority process (T1) and a high priority process (T4) where the high priority process (T4) and the low priority process (T1) share a given resource (SM4, 402'), characterized in that the method comprising the step of:

- temporarily raising an effective priority of the low priority process (T1) when the low priority process (T1) is going to use the shared resource (SM4, 402'), where the effective priority is raised to be above a priority of an other process (T1, T2) in the multiprocessing environment.

2. Method according to claim 1, characterized in that the step of raising the effective priority comprises the steps of:

- executing/assigning an additional process (T3, T5) accessing the shared resource (SM4, 402') on behalf of the low priority process (T1) where the additional process (T3, T5) has a priority equal to the effective priority, and
- where the additional process (T3, T5) is synchronised with the low priority process (T1).

3. Method according to claim 1, characterized in that the multiprocessor environment comprises a real-time operating system and a non-real time operating system running on a single processor at least at a given time, where the real-time operating system comprises said high priority thread (T4) and said additional process (T3, T5) and where the non-real time operating system comprises said low priority thread (T1).

4. Method according to claim 2, characterized in that

- the additional process (T3) and the low priority process (T1) are synchronised using a first semaphore (S1A) and a second semaphore (S1B).

5. Method according to claim 1, characterized in that the effective priority is raised at least until

- the low priority process (T1) has accessed or used the shared resource (SM4, 402'), or
- the high priority process (T4) has accessed or used the shared resource (SM4, 402') if the high priority process (T4) attempts to access or use the shared resource (SM4, 402') while the low priority process (T1) has access or uses the shared resource (SM4, 402').

6. Method according to claim 1, characterized in that the effective priority of the low priority process (T1) is raised to be slightly below that of the high priority process (T4).

7. Method according to claim 4, characterized in that access to the shared resource (SM4, 402') is controlled by a mutex (M) whereby said additional process (T3, T5) will not wait for the low priority process (T1) as long as it owns the mutex (M).

8. Method according to claim 1, characterized in that the shared resource (SM4, 402') is selected from the group of:

- a shared memory (SM4, 402'),
- a shared file (SM4, 402'), and
- a shared input/output (I/O) device.

9. Method according to claim 1, characterized in that the high priority process (T4) executes time-critical tasks.

10. A system (400) for executing processes with different priorities in a multiprocessing environment comprising means (401) adapted to execute a low priority process (T1) and a high priority process (T4) where the high priority process (T4) and the low priority process (T1) share a given resource (SM4, 402'), characterized in that the system comprises:

- means (401) for temporarily raising an effective priority of the low priority process (T1) when the low priority process (T1) is going to use the shared resource (SM4, 402'), where the effective priority is raised to be above a priority of an other process (T1, T2) in the multiprocessing environment.

11. System (400) according to claim 10, characterized in that the means (401) for raising the effective priority is conceived to:

- executing an additional process (T3, T5) accessing the shared resource (SM4, 402') on behalf of the low priority process (T1) where the additional process (T3, T5) has a priority equal to the effective priority, and
- where the system (400) comprises synchronisation means (401) conceived to synchronise the additional process (T3, T5) with the low priority process (T1).

12. System (400) according to claim 10, characterized in that the system comprises a single processor (401) comprising a real-time operating system and a non-real time operating system running on the single processor (401) at least at a given time, where the real-time operating system comprises said high priority thread (T4) and said additional process (T3, T5) and where the non-real time operating system comprises said low priority thread (T1).

13. A computer readable medium having stored thereon instructions for causing one or more processing units to execute the method according to any one of claims 1 to 9.

14. Set-top box comprising the system according to claim 10.

15. Television set comprising the system according to claim 10.